**Descrição do Problema**

Este projeto foi desenvolvido no âmbito da disciplina de Algoritmos e Estruturas de Dados (AEDA), com a vista na implementação de um sistema que simulasse a gestão de uma Seleção Nacional de Futebol.

Como é normal numa estrutura desta envergadura, temos essencialmente três tipos de pessoas: os jogadores que representam a seleção nas competições internacionais, os membros da equipa técnica e, por fim, os trabalhadores da classe mais baixa, que podem ir desde o jardineiro até à senhora da limpeza.

Tendo em vista sempre um alto nível de competição e desempenho pessoal, este sistema permite visualizar de forma minuciosa as estatísticas de cada jogador numa determinada convocatória, num determinado jogo, etc. Desta forma, conseguimos ter sempre uma noção geral não só de como está a ser a *performance* de cada jogador, mas também da equipa como um todo.

É neste aspeto que a nossa implementação pretende incidir. Estamos perante uma área onde todo o detalhe, ainda que ínfimo, é importante e pode fazer a diferença. Deste modo, apresentamos a *National Team*.

**Descrição da Solução**

Dado o problema apresentado acima, a nossa *approach* ao mesmo baseou-se na utilização das seguintes classes:

* Call
* Date
* Game
* GameStats
* IndividualStats
* InfCall
* NationalTeam
* Staff
* OtherWorker
* TechnicalTeam
* SoccerPlayer

Tentamos também fazer algum controlo de informação, utilizando algumas classes de

exceção, tais como:

* WrongDateFormat (membro da classe Date)
* NoObjectFound
* WrongStatistic

No que diz respeito ao utilizador, o mesmo não consegue aperceber-se desta forma de estruturação, uma vez que a única ligação que mantém com o programa é o menu.

O projeto baseia-se numa classe principal denominada NationalTeam. A mesma é o cerne de toda a estrutura pois contém toda a informação relativa a tudo. Por exemplo, contém vetores para os *Soccer Players*, *Other Workers*, *TechnicalTeam*, convocatórias (*Call)*, etc. Dentro de cada convocatória temos informações relativas à data chegada e saída de cada jogador. Esta situação é fácil de explicar pegando, por exemplo no caso da seleção Portuguesa.

Imaginemos que irá haver um estágio na Polónia, onde Portugal irá disputar alguns jogos amigáveis. A data de chegada ao país é dia 11 de fevereiro de 2020 e a data de saída é dia 20 de fevereiro de 2020. Como é frequente, a esmagadora maioria dos jogadores optará por seguir rigidamente estas datas.

No entanto, pensemos no caso do Cristiano Ronaldo. Pela altura da escrita deste relatório, já se encontra com 34 anos. Tal como todos sabemos um dia a alegria que ele nos dá irá acabar, portanto resta-nos tentar maximizar o desempenho dele. Desta forma, com a permissão da Seleção, o Cristiano chegará à Polónia apenas no dia 13 de fevereiro. É para isso mesmo que nós temos uma classe *InfCall* que regista todas essas datas de chegada e saída de cada jogador, aproximando-se assim do que de facto é praticado na realidade. Seguidamente, ainda dentro de cada convocatória, temos os vários jogadores pertencentes à mesma, bem como os jogos que irão decorrer entre as datas definidas na convocatória. Dentro de cada jogo, temos as estatísticas gerais da equipa principal, bem como da equipa adversária e também as estatísticas individuais de cada jogador. No ato da criação de uma *Call*, é verificada se a soma das estatísticas individuais de cada jogador, corresponde de facto às estatísticas globais da equipa principal. Quer isto dizer por exemplo que a soma dos remates efetuados pelo jogador A, B, etc… Terá de ser igual aos remates efetuados pela equipa, no total.

Para além disso, implementamos várias opções no menu que permitem a visualização quer de estatísticas individuais, quer da equipa, numa determinada *Call* ou *Game*. Temos também a opção de cálculo de custos para: *Soccer Player*, equipa e a *National Team* como um todo, num dado mês, ou desde sempre. Este cálculo de custos é baseado no atributo *daysActive*, presente na classe *SoccerPlayer* e também no seu preço de mercado (*marketPrice)*. Considera-se os dias ativos de um jogador como os dias em que já esteve ao serviço da seleção numa convocatória. Ou seja, a cada convocatória em que um jogador participa, o atributo *daysActive* será incrementado o número de dias que dura a convocatória. Para o cálculo dos custos, é feito o processo inverso. Utilizou-se um algoritmo de ordenação que permite pesquisar as convocatórias da mais recente para a mais antiga de modo a que desta forma, como o cálculo do seguro diário é calculado com base no *daysActive* que o jogador tinha no início da convocatória, se faça a substração do número de dias que dura convocatória para obter o *daysActive* utilizado para o cálculo da seguro diário.

Em caso de lesão do jogador na convocatória, esse valor triplicará. Foi também utilizado o algoritmo de pesquisa *binary search*, sobretudo no ato de procura da existência de um dado *Soccer Player* numa determinada convocatória.

Por fim, relativamente a funcionalidades adicionais, implementamos duas opções que permitem saber quantas vitórias, derrotas e empates a seleção tem, bem como o saber qual foi o *top scorer*!

**UML – Diagrama de Classes**

**A screenshot of a cell phone

Description automatically generated**

**<asd**

Nota: O diagrama será enviado em conjunto com o relatório para uma melhor visualização do mesmo.